Call REST API from ASP.NET Core Blazor

In this video we will discuss **how to call and consume a REST API from ASP.NET Core Blazor application**.

Can a Blazor component call REST API directly

Yes, a Blazor component can directly call a REST API. However, for separation of concerns and to keep the component code clean, it's a good practice to create a separate service that calls the REST API.

Create a service to call REST API

Add a folder with name *Services* to the Blazor web application project. Add the following 2 class files to this folder.

1. IEmployeeService.cs
2. EmployeeService.cs

IEmployeeService.cs

public interface IEmployeeService

{

Task<IEnumerable<Employee>> GetEmployees();

}

Copy

EmployeeService.cs

* We are using HttpClient class to call the REST API service.
* This class is in System.Net.Http namespace.
* HttpClient is injected into the EmployeeService using dependency injection.
* We have not registered HttpClient service with the dependency injection container yet. We will do that in just a bit.
* We are using httpClient.GetJsonAsync to call the REST API. This method is in Microsoft.AspNetCore.Blazor.HttpClient Nuget package. Install this package and do not forget to include the namespace Microsoft.AspNetCore.Components.
* Pass the REST API endpoint (api/employees) to httpClient.GetJsonAsync method.

httpClient.GetJsonAsync<Employee[]>("api/employees")​

Copy

using EmployeeManagement.Models;

using Microsoft.AspNetCore.Components;

using System.Collections.Generic;

using System.Net.Http;

using System.Threading.Tasks;

namespace EmployeeManagement.Web.Services

{

public class EmployeeService : IEmployeeService

{

private readonly HttpClient httpClient;

public EmployeeService(HttpClient httpClient)

{

this.httpClient = httpClient;

}

public async Task<IEnumerable<Employee>> GetEmployees()

{

return await httpClient.GetJsonAsync<Employee[]>("api/employees");

}

}

}

Copy

Register HttpClient Services

In ConfigureServices method of the Startup class register HttpClient Services using AddHttpClient method.

public void ConfigureServices(IServiceCollection services)

{

services.AddRazorPages();

services.AddServerSideBlazor();

services.AddHttpClient<IEmployeeService, EmployeeService>(client =>

{

client.BaseAddress = new Uri("https://localhost:44379/");

});

}

Copy

Call Service from Blazor Component

* Finally call IEmployeeService from EmployeeList blazor component.
* We use [Inject] attribute to inject a service into a Blazor component. We cannot use a constructor for this.
* In the component OnInitializedAsync method, we call the EmployeeService.GetEmployees method.
* The data (list of employees) that this method returns is then used to initialise Employees property.
* The EmployeeList blazor component binds to this Employees property to display the list of employees.

using EmployeeManagement.Models;

using EmployeeManagement.Web.Services;

using Microsoft.AspNetCore.Components;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

namespace EmployeeManagement.Web.Pages

{

public class EmployeeListBase : ComponentBase

{

[Inject]

public IEmployeeService EmployeeService { get; set; }

public IEnumerable<Employee> Employees { get; set; }

protected override async Task OnInitializedAsync()

{

Employees = (await EmployeeService.GetEmployees()).ToList();

}

}

}